# 4.JavaScript

## 4.1.JavaScript概述

**JavaScript是干什么的？**

1. HTML 定义了网页的内容

2. CSS 描述了网页的布局

3. JavaScript 指定了网页的行为

**Js历史**

1995年5月，网景公司，叫Brendan Eich的哥们，10天，LiveScript

1995年12月，改名为JavaScript

1996年8月，微软，Jscript

1997-1999年，ECMA国际组织，ECMAScript，基于已有的JavaScript和Jscript，提出了标准的Script语法规则，JavaScript和Jscript都遵循这套标准。

1999-以后，ECMAScript不断的更新

与java的关系？

不好意思，只是蹭了下你的热度，我们其实没关系

**js特点**

JavaScript 是脚本语言，不需要编译，是解释运行的语言。

JavaScript 是可插入 HTML 页面的编程代码，可由所有的现代浏览器执行。

JavaScript 是基于对象，弱类型的语言。

JavaScript 很容易学习。

**js优点：**

交互性：可以与用户进行动态交互。

安全性：只能在浏览器内运行，不能访问本地硬盘或其他资源。

跨平台：有浏览器即可运行，与操作环境无关。

## 4.2.在html中引入js代码

（案例一）

**1.直接在html中书写js代码**

在html中可以在script标签内部直接书写js代码，通常写在head中！

**2.引入外部的js文件**

如果js代码特别多的情况下, 可以将js代码抽取到一个独立的文件中, 在html中通过script标签引入该js文件.

注意: script标签如果是用来引入外部的js文件, 标签内部不要再书写js代码, 并且标签不要自闭, 否则可能会导致引入失败！！！

**案例一：在html中引入js**

|  |
| --- |
| <!-- 第一种方式 -->  <script type="text/javascript">  function fn1() {  alert("引入js的第一种方式")  }  </script>  <!-- 第二种方式（先创建demo.js文件，里面直接写js代码） -->  <script type="text/javascript" src="demo.js"></script> |

## 4.3.JavaScript语法

### 4.3.1.注释

单行注释: // 注释内容

多行注释: /\* 注释内容 \*/

### 4.3.2.数据类型

**基本数据类型**

* 数值类型(number)
* 字符串类型(string)
* 布尔类型(boolean)
* undefined类型
* null类型

**（1）数值类型**

在js中, 所有的数值底层都是浮点型¸ 在需要时, 整型和浮点型会自动的进行类型的转换.

例如：2.4+3.6=6

特殊值：

Infinity 正无穷大

-Infinity 负无穷大

NaN (not a number) 非数字 ：NaN和任何数值都不相等, 包括它本身. 如果要判断一个值是否是非数字, 不能用 NaN == xx 来判断, 而是应该使用 isNaN(xx) 进行判断。

在js中, 为number类型提供了对应的包装对象 – Number。

**（2）字符串类型**

在js中, 字符串类型是基本数据类型的一种, 字符串常量, 可以用单引号或者双引号引起来！

例如：var s1 = “aaa”; var s2 = ‘bbb’;

在js中, 为string类型提供了对应的包装对象 – String。

**（3）布尔类型**

boolean, 值为true或false

在js中, 为boolean类型提供了对应的包装对象 – Boolean。

**（4）undefined**

undefined类型的值只有一个, 就是undefined. 表示变量未定义。如果声明了一个变量, 但是没有初始化值, 该变量的值就是undefined

例如：var a; 则a的值即为undefined。

**（5）null**

null类型的值也只有一个, 就是null, 表示此处的值现在为空。常用来作为函数的返回值, 表示该返回值是一个没有任何内容的空对象

**复杂数据类型**

复杂数据类型: 对象(普通对象 数组 函数)（后面详讲）

**js中的数据类型的转换（案例二）**

js中的数据类型在需要时会自动的进行类型转换, 转换时遵循如下规则:

**数值类型:**

转字符串类型, 直接转成对应值的字符串, 3 --> "3"

转布尔类型, 0和NaN转成false, 其他数值转成true

在需要时, 会自动转成对应的包装对象 100 --> new Number(100)

**字符串:**

空字符串(""): 转数值为0, 转布尔值为false

非空纯数值字符串("123"): 转数值为对应的数值, 转布尔值为true

非空非数值字符串("abc"): 转数值为NaN, 转布尔值为true

在需要时, 会自动转成对应的包装对象. "aaa" --> new String("aaa")

**布尔类型:**

true: 转数值为 1, 转字符串为"true"

false: 转数值为 0, 转字符串为"false"

在需要时, 会自动转成对应的包装对象.

**undefined**

转数值为NaN, 转字符串为"undefined", 转布尔值为false, 转对象会抛出异常!

**null**

转数值为 0, 转字符串为"null", 转布尔值为false, 转对象会抛出异常!

**案例二：数据类型转换**

|  |
| --- |
| <!-- 数据类型转换 -->  <script type="text/javascript">  //1.数值类型  //1.1.弹出数字6与字符串"66"的和  alert(6+"66");  //1.2.对数字进行判断，若为0或NaN，则弹出"世界如此美好！" 若为其他数字，则弹出"我却如此暴躁！"  if(3){  alert("我却如此暴躁！");  }else{  alert("世界如此美好！");  }  //2.字符串  //2.1.弹出字符串"1000"与数字334的差  alert("1000"-334);  //2.2.字符串判断，若为空字符串("")，弹出"师傅，大师兄说的对！"，若为非空字符串，弹出"大师兄，师傅被妖怪抓走了！"  if("123"){  alert("大师兄，师傅被妖怪抓走了！");  }else{  alert("师傅，大师兄说的对！");  }  //3.布尔类型  /\* 已知：var a = [3>2>1, 1<2<3] 则a[0]和a[1]的值分别是什么？\*/  var a = [3>2>1, 1<2<3];  alert(a[0]); //false  alert(a[1]); //true  </script> |

### 4.3.3.变量的定义

在js中有数据类型, 但是变量不区分类型, 所以称js是一门弱类型的语言。

在js中通过var关键字声明变量, 变量不区分类型, 可以指向任意类型的值!

例如：var s1 = “aaa”; var i = 100;

### 4.3.4.运算符

（案例三）

JS中的运算符和Java中的运算符大致相同

JS中与java运算符不同之处：

比较运算符除了“==”，还有“===”

== 在进行比较时, 如果两端的值不是同一种数据类型, 先自动转成同一种数据类型, 再进行比较, 如果转换之后再比较, 相等则返回true, 不相等则返回false;

=== 为严格相等，在进行比较时, 如果两端的值不是同一种数据类型, 直接返回false; 如果是同一种类型再次比较值是否相等, 是则返回true, 不是则返回false

typeof: 用来返回变量/常量/表达式的数据类型

格式：var x = 123; typeof x

delete: 用来删除数组中的元素或者删除对象中的属性或方法。。。

格式：var arr = [123,”abc”,true]; delete arr[1];

**案例三：运算符**

|  |
| --- |
| <!-- 运算符演示 -->  <script type="text/javascript">  //1.分别用"=="和"==="判断数字666和字符串"666"是否相等  alert(666=="666"); //true  alert(666==="666"); //false  //2.分别对x进行不同的赋值，利用typeof查看x的数据类型  var x = "1";  alert(typeof x);  //3.问题：判断typeof 6+"66" 的值是什么？  alert(typeof 6+"66");//number66  //4.删除下面数组中的第二个元素  var arr = [111,"abc",true];  delete arr[1];  alert(arr);  alert(arr[1]);  </script> |

### 4.3.5.语句

JS中的语句和Java中的语句也大致相同

if条件语句

如果判断条件不是布尔类型, 会自动转换成布尔值!

switch语句

语法和Java中大致相同, 并且也支持字符串类型

循环语句

while(){} do{}while() for(){} 语法和Java中的很相似, 但是js中不支持增强for循环

### 4.3.6.函数

（案例四）

**3.6.1.通过function关键字来定义一个函数**

function 函数名(形参列表){

函数体

}

调用函数: 函数名(实参列表)

**3.6.2.通过函数表达式来定义一个函数**

var fn = function (形参列表){

函数体

}

调用函数: 函数名(实参列表)

**案例四：函数**

|  |
| --- |
| <!-- 函数演示 -->  <script type="text/javascript">  //1.分别用两种方式定义一个方法，将姓名与年龄作为参数传入，调用方法时弹出姓名与年龄  //方式一  function fn1(name,age){  alert(name+":"+age);  }  fn1("兰姐",18);  //方式二  var fn2 = function(name,age){  alert(name+":"+age);  }  fn2("刚哥",19);  </script> |

### 4.3.7.数组

**通过Array()构造函数来创建数组**

var arr1 = new Array(); //创建一个空数组

var arr2 = new Array(10); //创建一个长度为10的空数组

var arr3 = new Array(“aa”, 55, true); //创建一个具有指定初始值的数组

**通过数组直接量来创建数组**

var arr4 = []; //创建一个空数组

var arr5 = [“x”, 100, false]; //创建一个具有指定初始值的数组

**数组中的细节问题:**

>> 数组的长度可以被任意改变

var arr =[“aa”, “bb”, “cc”]; arr.length=1; //[“aa”]

>> 如果数组中某一个位置没有元素, 该位置的值为undefined

var arr =[“aa”, “bb”, “cc”]; arr[10]; //undefined

>> JS中的数组可以存放任意类型的数据

var arr = [“x”, 100, false];

## 4.4.js中的内置对象

（案例五）

### 4.4.1.String对象

创建：

var str1 = new String(“xxx”); //创建字符串对象

var str2 =“xxx”; //创建字符串基本类型，但是也可以

常用属性和方法：

str.length 返回字符串长度。

str.charAt(index) 返回指定索引位置处的字符。

例如：var str = "abcdeabc"; str.charAt(3) // d

str.indexOf(subString, startIndex) 返回第一次出现子字符串的位置。

例如：var str = "abcdeabc"; x.indexOf("bc",2) // 6，从第3个开始数

str.lastIndexOf(substring, startindex) 返回最后出现子字符串的位置。

例如：var str = "abcdeabc"; x.lastIndexOf("bc",8); // 6，从后往前数

str.split(delimiter) 将字符串按照指定字符分割为字符串数组。

例如：var str = "a:b:c:d"; str.split(":") //返回为一个数组[a,b,c,d]

str.slice(start, end) 提取字符串的某个部分，含头不含尾。

例如：var str = "abcdeabc"; str.slice(2,4); // cd

str.substr(start, length) 返回从指定位置开始的指定长度的子字符串。

例如：var str = "abcdeabc"; str.substr(2,4); //cdea

str.toLowerCase( ) 字符串中的字母转换为小写。

例如：var str = "ABCDE"; str.toLowerCase(); //abcde

str.toUpperCase( ) 字符串中的字母转化为大写。

例如：var str = "abcde"; str.toUpperCase(); //ABCDE

**str.match(regexp) 在字符串中查找指定匹配正则表达式的值。**

**str.replace(regexp, replaceText) 字符串中匹配正则的值替换为其他值。**

**str.search(regexp) 查找与正则表达式匹配的第一个子字符串的位置。**

### 4.4.2.RegExp 对象（正则表达式）

创建：

var reg1 = new RegExp("", "");

var reg2 = /xxx/;

标识符: g 🡪 Global 全局查找 i 🡪 IgnoreCase 忽略大小写

如果正则需要从头到尾都匹配，需要以”^”开头，以”$”结尾。

reg.test(str); 检查字符串是否符合该正则, 如果符合返回true, 不符合返回false！！！

### 4.4.3.Array对象

创建：

var arr1 = new Array();

var arr2 = [];

常用属性和方法：

arr.length 返回数组中元素的个数

例如：var arr = ["a","b","c"]; arr.length; // 3

arr.concat(arr1,arr2,arr3...) 将多个数组合并成一个

例如：

var arr1 = ["a","b","c"]; var arr2 = ["d","e","f"]; var arr3 = ["x","y","z"];

arr1.concat(arr2,arr3); // ["a","b","c","d","e","f","x","y","z"]

arr.join(string) 将数组中所有元素按照指定字符串连接成一个字符串

例如：var arr = ["aaa",123,true]; arr.join("-"); // aaa-123-true

arr.reverse( ) 返回一个元素顺序被反转的 Array 对象。

例如：var arr = ["a","b","c"]; arr.reverse(); //["c","b","a"]

arr.slice(start, end) 返回一个数组的一段，含头不含尾。

例如：var arr = ["a","b","c",1,2,3]; arr.slice(2,4); // ["c",1]

arr.splice(start, deleteCount,value...)

从一个数组中移除一个或多个元素，如果必要，在所移除元素的位置上插入新元素，返回所移除的元素。

例如：var arr = ["a","b","c","d","e","f"]; arr.splice(2,2,"zzz");

// 返回值为["c","d"]，arr则变为["a","b","zzz","e","f"]，中间被替换

arr.pop( ) 移除数组中的最后一个元素并返回该元素

例如：var arr = ["aaa",123,"x",true,8]; arr.pop();

//返回值为 8，arr则变为["aaa",123,"x",true]

arr.push(value) 在数组的末尾加上一个或多个元素，并返回新数组的长度值

例如：var arr = ["aaa",123,"x",true,8]; arr.push("last")

//返回值为6，arr则变为 ["aaa",123,"x",true,8,"last"]

arr.shift( ) 移除数组中的第一个元素并返回该元素

例如：var arr = ["aaa",123,"x",true,8]; arr.shift()

//返回值为 "aaa" ，arr则变为[123,"x",true,8]

arr.unshift(value) 为数组的开始部分加上一个或多个元素,并且返回该数组的新长度

例如：var arr = ["aaa",123,"x",true,8]; arr.unshift("first")

//返回值为6，arr则变为["first","aaa",123,"x",true,8]

**arr.sort(sortfunction) 返回排序后的数组。**

### 4.4.4.date对象

创建：

var date1 = new Date(); // 当前时间

var date2 = new Date(年,月,日[,时,分,秒]); //指定时间，可以只有年月日

注意：当创建指定时间的时候，月是从0开始，例如：

var date2 = new Date(2008,7,8,20,0,0);

这个时间是2008年8月8日晚上8点整

常用方法：

data.toLocaleString() 把日期(一个数值)转变成本地日期格式字符串

例如：var date2 = new Date(2008,7,8); date2.toLocaleString();

// "2008/8/8 上午12:00:00"

data.[getFullYear()](http://www.w3cschool.cn/tryrun/showhtml/tryjsref_getfullyear" \t "http://www.w3cschool.cn/javascript/_blank) 获取日期对象中所表示时间的年份

例如： var date = new Date(2008,7,8); date.getFullYear(); // 2008

data.[getMonth()](http://www.w3cschool.cn/tryrun/showhtml/tryjsref_getfullyear" \t "http://www.w3cschool.cn/javascript/_blank) 获取日期对象中所表示时间的月份，返回一个 0 到 11 之间的整数，0表示一月。

例如：var date = new Date(2008,7,8); date.getMonth(); // 7

data.[getDate()](http://www.w3cschool.cn/tryrun/showhtml/tryjsref_getfullyear" \t "http://www.w3cschool.cn/javascript/_blank) 获取日期对象中所表示一个月的日期值，返回值是一个处于 1 到 31 之间的整数，代表了相应的日期值。

例如：var date = new Date(2008,7,8); date.getDate(); // 8

data.getDay() 获取日期对象中所表示一个周的日期值。 返回的值是一个处于 0 到 6 之间的整数，0表示周日，6表示周六。

例如：var date = new Date(2008,7,8); date.getDay(); // 5，表示周五

data.getTime() 返回从 1970 年 1 月 1 日至指定时间的毫秒数。

例如：var date = new Date(2008,7,8); date.getTime(); //1218124800000

### 4.4.5.Math对象

Math可以直接拿来用！

常用属性和方法：

Math.PI 返回圆周率的值，约等于 3.141592653589793。

例如：var num = Math.PI; // num 的值为3.141592653589793。

Math.exp(number) 返回 e（自然对数的底）的幂。

例如：Math.exp(1) // e1 约等于2.718281828459045

Math.abs(number) 返回数字的绝对值。

例如：Math.abs(-5) // 5

Math.ceil(number) 向上取整

例如：Math.ceil(3.14) // 4

Math.floor(number) 向下取整

例如：Math.floor(3.14) // 3

Math.round(number) 四舍五入到整数

例如：Math.round(5.67) // 6

**Math.random() 返回介于 0 和 1 之间的伪随机数。**

**案例五：js内置对象**

|  |
| --- |
| <!-- js内置对象 -->  <script type="text/javascript">  //一、字符串String  //1.将字符串中所有的"java"提取到一个数组中  var str = "zhang,java,wang,JAVA,li,java,zhao";  var reg1 = /java/g;  alert(str.match(reg1));  //2.将字符串中所有的"java"不分大小写提取到一个数组中  var reg2 = /java/ig;  alert(str.match(reg2));  //3.将所有的"java"不分大小写，全部换成"C++"  alert(str.replace(reg2,"C++"));  //4.查找第一个"java"出现的位置(不分大小写)  alert(str.search(reg2));  //二、正则  //检查str字符串是否符合如下正则  var reg3 = /wang/;  var reg4 = /^wang$/;  alert(reg3.test(str));  alert(reg4.test(str));  //三、数组  //对arr进行排序，思考排序后的顺序  var arr = ["aaa",123,"x",true,8];  alert(arr.sort());    //四、Math对象  //1.返回一个1-100之间的随机整数  alert(Math.ceil(Math.random()\*100));  //2.返回一个30-50直接的随机数  alert(Math.round(Math.random()\*20+30));  </script> |

### 4.4.6.全局对象

全局对象没有语法。直接调用其方法。

parseFloat(numString) 将字符串转换成浮点数。

例如：parseFloat("13.14") // 返回number类型的13.14

parseInt(numString) 将字符串转成整数，非四舍五入。

例如：parseInt("9.99") // 返回number类型的9

isNaN(numValue) 判断一个值是否为非数字。

例如： isNaN("abc") // true

isNaN("123") // false

eval(codeString) 判断一个字符串并将其以脚本代码的形式执行

例如：eval("alert(1+2)"); //会直接将"alert(1+2)"当作代码执行，弹出3

## 4.5.自定义对象

（案例6）

**方式一：构造函数定义对象**

function Person(){} // 定义一个空的Person对象

function Person2(参数1,参数2...){

对象内容...

} //定义一个带参数的对象

**方式二：对象直接量**

var p3 = {

"key1" : "value1",

"key2" : "value2".........

}

**对象中取值：**

P3.key1 或者 p3["key1"]

**案例六：自定义对象**

|  |
| --- |
| <!-- 自定义对象 -->  <script type="text/javascript">  //1.分别用两种方式定义一个对象，包含姓名和年龄，以及弹出姓名年龄的方法  //方式一  function Person1(){};  var p1 = new Person1();  p1.name = "尼古拉斯赵四";  p1.age = 1;  p1.run = function(){  alert(this.name+":"+this.age)  }  p1.run();    function Person2(name,age){  this.name = name;  this.age = age;  this.run = function(){  alert(this.name+":"+this.age);  }  }  var p2 = new Person2("尼古拉斯赵四",30);  p2.run();  //方式二  var p3 = {  "name" : "尼古拉斯赵四",  "age" : "80",  "run" : function(){  alert(this.name+":"+this.age);  }  }  p3.run();  </script> |

## 4.5.DHTML

### 4.5.1.DHTML概述

DHTML: Dynamic HTML 动态的 html

DHTML并不是一门新的技术, 而是将现有的HTML, CSS, JavaScript整合在一起, 形成了DHTML技术.

DHTML分为BOM和DOM。

BOM：Browse Object Modle 即浏览器对象模型，其中封装了浏览器操作相关对象。

DOM：Document Object Modle 即文档对象模型，将整个HTML文档按照文档结构组织成了树形结构。

### 4.5.2.BOM

（案例7）

**Window 对象**

此对象为全局对象，因此，调用方法或属性时，可以省略window。

常用事件：

onclick事件 – 当窗口被点击时触发

onfocus事件 – 当窗口获得焦点时触发

onblur事件 – 当窗口失去焦点时触发

onload事件 – 当整个html文档加载完之后立即触发

常用方法：

alert() – 消息对话框

confirm() – 确认对话框

**location对象**

属性：

href – 获取或设置浏览器地址栏url。。。

**案例7：BOM**

|  |
| --- |
| <!-- BOM -->  <script type="text/javascript">  //window对象  //1.在按钮点击时弹出提示框  function clickfn(){  alert("按钮被点击");  }  //2.输入框1获得焦点时弹出提示框  function focusfn(){  alert("输入框1获得焦点 ");  }  //3.输入框2失去焦点时弹出提示框  function blurfn (){  alert("输入框2失去焦点");  }  //4.弹出输入框3中的value值  onload=function(){  //获取输入框3的input按钮  var inp = document.getElementById("input3");  alert(inp.value);  }  //5.点击删除商品按钮，弹出提示是否删除,如果确定，则弹出商品已删除，否则弹出商品还在  function delfn(){  if(confirm("确认删除商品？")){  alert("商品已删除");  }else{  alert("商品还在");  }  }  //location对象  //1.弹出当前url  alert(location.href);  //2.将页面url设置为"http://www.baidu.com"  alert("此页面将跳转到tmooc");  location.href = "http://www.baidu.com";  </script> |

### 4.5.3.DOM

（DOM案例）

**获取元素**

document.getElementById("id值"); 通过id值获取指定id的元素

document.getElementsByName("name属性值"); 通过name属性获取指定name值的所有元素组成的集合数组

document.getElementsByTagName("元素名"); 通过元素的名称获取指定元素名的所有元素组成的集合数组

value属性： 获取或设置输入框的value值

innerText： (部分浏览器不支持) 获取或设置元素内的文本内容

innerHTML： 获取或设置元素的html内容

**元素的增删改**

docuemnt.createElement("div"); 创建一个div元素

body.appendChild(oDiv); 往body中追加一个子元素

body.removeChild(oDiv); 删除oDiv子元素

body.replaceChild(oNewDiv, oDiv); 用oNewDiv替换已有的子元素oDiv

body.insertBefore(oDiv1, oDiv); 往body中插入一个子元素oDiv1, 子元素是插入在oDiv的前面

oDiv.cloneNode([boolean]); 克隆oDiv元素, 参数默认值为false, 表示只克隆元素本身, 不克隆oDiv内的所有子孙元素, 如果希望克隆元素还克隆元素内容所有子孙元素, 可以传入参数true！！！

**Dom案例一**

|  |
| --- |
| /\* --通过ID获取并弹出用户名输入框的值-- \*/  function demo1(){  //根据ID获取用户名输入框，un即代表整个输入框  var un = document.getElementById("username");  alert(un.value);  //可以自己设置value值  un.value = "兰姐";  }    /\* --通过name属性获取并弹出密码输入框的值-- \*/  function demo2(){  //根据name属性获取密码输入框，因为name属性可以重复，所以获取的是一个集合数组  var pwarr = document.getElementsByName("password");  //在数组中获取第一个元素  var pw = pwarr[0];  alert(pw.value);  }  /\* --通过元素名称获取并弹出确认密码输入框的值-- \*/  function demo3(){  //根据标签名获取确认密码输入框，同样，获取的是一个数组  var inparr = document.getElementsByTagName("input");  //获取第三个元素  var pw2 = inparr[2];  alert(pw2.value);  }  /\* --获取元素内容-- \*/  function demo4(){  var oP = document.getElementById("pid");  //获取p标签中的文本内容 innerText(部分浏览器不支持)  alert(oP.innerText);  //获取p标签中的所有html内容 innerHTML(绝大部分浏览器都支持)  alert(oP.innerHTML);  //设置p标签中的html内容  oP.innerHTML = "<font color='blue'>我要变身了...!</font>";  } |

**Dom案例二**

|  |
| --- |
| /\* --添加节点: 添加一个div元素到body的最后面-- \*/  function addNode(){  //获取父元素  //var oBody = document.getElementsByTagName("body")[0];  //body是document的子对象，所以可以直接获取  var oBody = document.body;  //创建一个游离div元素  var oNewDiv = document.createElement("div");  //给div中添加内容  oNewDiv.innerHTML = "我是新来的....!";  //将div元素挂载到父元素上  oBody.appendChild(oNewDiv);  }  /\* --删除节点: 删除body中id值为 div\_3 的div元素-- \*/  function deleteNode(){  //获取父元素body  var oBody = document.body;  //获取要删除的元素div\_3  var oDiv\_3 = document.getElementById("div\_3");  //解除父子关系  oBody.removeChild(oDiv\_3);  }  /\* --更新节点: 用新节点替换id值为 div\_2 的div元素 -- \*/  function updateNode(){  //创建一个新节点  var oNewDiv = document.createElement("div");  oNewDiv.innerHTML = "我是来替换的~!";  //获取被替换的元素  var oDiv\_2 = document.getElementById("div\_2");  //获取父元素, 并通过父元素替换子元素  var oBody = oDiv\_2.parentNode;  oBody.replaceChild(oNewDiv, oDiv\_2);  }  /\* --克隆节点、插入节点到指定元素的前面--\*/  function copyNode(){  //获取指定元素  var oDiv\_4 = document.getElementById("div\_4");  //克隆元素  var oCloneDiv = oDiv\_4.cloneNode(true);  //false 默认值, 只克隆元素本身, 不克隆元素内容  //获取父元素body  var oBody = document.body;  var oDiv\_2 = document.getElementById("div\_2");  //将克隆元素插入到指定位置  oBody.insertBefore(oCloneDiv,oDiv\_2);  } |

**Dom案例三**

|  |
| --- |
| /\* --通过js实现访QQ列表好友分组-- \*/  function openDiv(thisobj){  var oDiv = thisobj.parentNode.getElementsByTagName("div")[0];  /\*  当点击一个分组时, 如果这个分组是展开的, 点击之后应该关闭当前分组  当点击一个分组时, 如果这个分组是关闭的, 点击之后应该先关闭所有, 再展开当前分组  \*/    if(oDiv.style.display == "block"){//分组是展开的  oDiv.style.display = "none";  }else{//分组是关闭的  //获取所有的分组, 依次关闭分组  var arrDiv = document.getElementsByTagName("div");  for(var i=0; i<arrDiv.length; i++){  arrDiv[i].style.display = "none";  }  //将div由隐藏设置为显示(展开分组)  oDiv.style.display = "block";  }  } |

**Dom案例四**

|  |
| --- |
| var data = {  "北京市": ["海淀区","朝阳区","丰台区"],  "河北省": ["石家庄","唐山","秦皇岛"],  "辽宁省": ["沈阳","大连","鞍山"],  "山东省": ["青岛","济南","烟台"]  }    //练习1  function getCountry(thisobj){  //当选项切换时, 触发onchange事件, 执行getCountry函数, 在函数中获取当前选中的国家并弹出!  alert(thisobj.value);  }    //练习2  function selectCity(thisobj){  //1.获取用户选中的省份  var prov = thisobj.value;//thisobj-->省份的select  //2.根据省份获取省份对应的城市列表  var citys = data[prov];  //3.将省份对应的所有城市作为option选项依次填充到城市select中  //>>获取城市select  var oCitySele = document.getElementById("city");  //>>在填充城市列表数据之前, 先清空之前的数据  oCitySele.innerHTML = "<option>--选择城市--</option>";    //填充城市列表数据  for(var i=0; i<citys.length; i++){  var opt = document.createElement("option");  opt.innerHTML = citys[i];  oCitySele.appendChild(opt);  }  } |

## 4.6.作业练习

1. javascript是一种什么样的语言

答：解释性脚本语言，代码不进行预编译

主要用来向HTML页面添加交互行为

可以直接嵌入HTML页面，也可单独写成js文件并引入

跨平台性，在绝大多数浏览器支持下，可以在多种平台下运行，linux,windows

1. == 和 === 区别

答：前者会自动转换类型，再进行比较，后者不会转换类型，如果类型不同直接返回false

3.添加 删除 替换 插入到某个节点的方法

答：添加：obj.appendChild(childNode)

删除：obj.removeChild(childNode)

替换：obj.replaceChild(newChild,oldChild)

插入：obj.innersetBefore(newChild,oldChild)